

RESEARCH

Open Access

Secure ubiquitous authentication protocols for RFID systems

Md Monzur Morshed*, Anthony Atkins and Hongnian Yu

Abstract

In this article, the possible privacy and security threats to the radio frequency identification (RFID) systems are investigated and new authentication protocols are proposed which provide the identified privacy and security in a very efficient manner for a ubiquitous computing environment. The approach utilizes the concepts of two very different, widely known RFID protocols, i.e. the “low-cost authentication protocol (LCAP)” approach and the “one-way hash-based LCAP” approach. The resulting protocols combine the advantages of both protocols and eliminate the problems from these. The approaches are evaluated using a variety of criteria that are relevant in practice. The proposed protocols use random numbers and a hash function to encrypt the key to protect the RFID system from the adversary attacks. The hash value is used as a hash address to reduce the search time to locate the tag in the database from a large number of records. A simulation experiment is conducted to verify some of the privacy and security properties of the proposed protocols.

Keywords: RFID, security, authentication protocol, ubiquitous, hash address

1. Introduction

Radio frequency identification (RFID) tags emerge as the successor of barcodes and are used in many applications such as in automation of automobiles, animal tracking, highway toll collection and supply-chain management [1]. An RFID tag has some advantages over an optical barcode that makes it more suitable in automation. A barcode indicates the type of the object on which it is printed but the RFID tag gives a unique serial number that distinguishes the object uniquely from many millions of similar types of products. Another advantage of an RFID tag is that it does not require line-of-sight contact with the readers as in optical barcodes. RFID is a technology to identify objects or people automatically [2]. An RFID system consists of three components: tag, reader and back-end database [3]. It is a small and extremely low-priced device consists of a microchip with very limited functionality and data storage and an antenna for wireless communication with readers. An RFID tag can be passive or active depending on the powering technique. In general, passive tags are inexpensive. They have no on-board power; they get power from the signal of the

interrogating reader. Active tags contain batteries to power their transmission. RFID readers with antennas are devices used to read or write data from or to the RFID tags. The readers send query to a tag to obtain information from the tag. The database stores the information about the tags and the readers [4].

The RFID tag in the form of electronic product code (EPC) tag is the most popular standard and is specified by an organization called EPCglobal Inc. [5]. An EPC tag traditionally contains some information such as a product type identifier, a manufacturer identifier and a unique serial number those are exposed to the reader. This unique serial number works as a unique identifier (*ID*). Due to this unique serial number in an RFID tag, it is possible to track the tag uniquely. Due to this, the information in an RFID system is vulnerable to unauthorized readers. An RFID system is vulnerable to various attacks such as eavesdropping, traffic analysis, spoofing and denial of service. These attacks may reveal sensitive information of tags and hence break a person's privacy. Another type of privacy violation is traceability which establishes a relation between a person and a tag. If a link can be established between a person and the tag, the tracing of the tag makes the tracing of the person possible [1]. To protect the privacy in an RFID system, a tag needs to authenticate a

* Correspondence: m.m.morshed@staffs.ac.uk
Faculty of Computing, Engineering and Technology, Staffordshire University,
Stafford, UK

reader. However, it is infeasible to use conventional cryptography in a passive RFID tag due to its extremely limited processing and memory limitations. A typical RFID system is shown in Figure 1.

The objective of the article is to propose new efficient and effective protocols to address these issues for RFID systems in ubiquitous computing environment. The protocols are based on the challenge-response method using a one-way hash function, a static identifier and two random numbers in the RFID systems. The purpose of the hash function is to give a one-way hash result so that an adversary cannot extract the input from the output. The value of the hash function is also used as a hash address for the tags in the database. The purpose of the random numbers is to make the response anonymous. This protocol protects the privacy and security of RFID systems of the issues outlined above.

The rest of the article is organized as follows. In Section 2, the privacy and security model in RFID system is discussed. Also the performance criteria of the RFID systems are discussed in this section. Section 3 contains the related studies. In Section 4, contributions and protocols are presented. Section 5 outlined the privacy, security and efficiency analysis for evaluation of our protocols. Section 6 describes the simulation result and evaluation. Section 7 concludes the article citing major contributions.

2. Privacy and security in RFID systems

The privacy and security objective in RFID system is to protect the communication between the reader and the

tag from various attacks. We identify the following privacy and security issues:

- **Information leakage:** In a typical RFID system, a tag has a unique identifier that is transmitted to the reader. So it can easily be identified with this unique serial number. Due to this unique serial number, the information in it is vulnerable to an adversary. For the protection from information leakage, an RFID system needs to provide privacy control so that unauthorized readers cannot access the tags.

- **Traceability and location privacy:** If the response of a tag can be linked to the tag then the location of the tag can be tracked. If a tag transmits a static response to a reader, an adversary can distinguish it from other responses. If the responses from the tags are anonymous, then the tracking problem can be avoided.

- **Impersonation and replay attack:** An adversary can query to a tag or a reader and can impersonate the tag or the reader. If an adversary can collect the information during communication from the tag and the reader they can impersonate the tag to explore more information. An adversary can use this information and replay in the future.

- **Denial of service (DoS):** An adversary may disrupt the communication between a valid reader and a tag. If the adversary can successfully block the transmission it can cause the server and the tag to lose synchronization. The RFID system should be able to handle this to keep the synchronization of the tag and the reader.

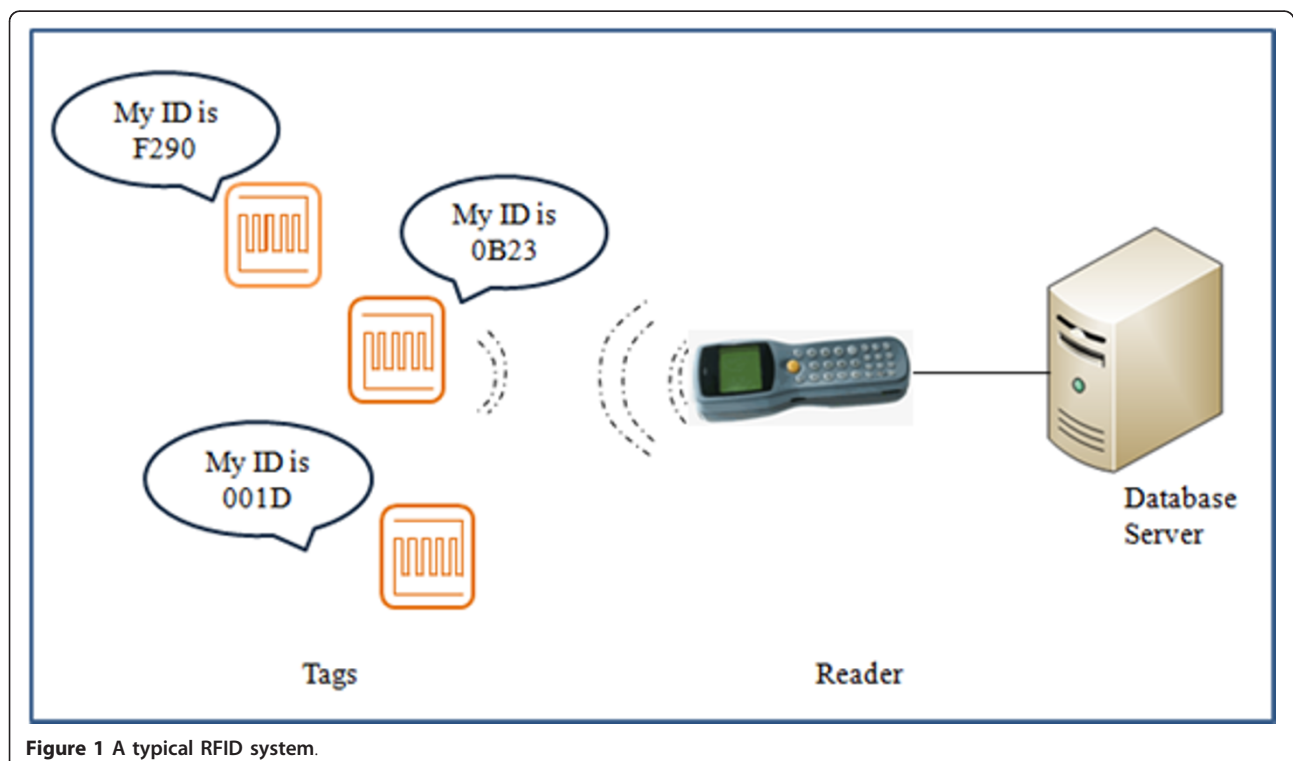


Figure 1 A typical RFID system.

3. Related studies

There are many approaches for the privacy and security of the RFID systems. Sharma et al. [6] mentioned about the resource constrained in an RFID tag as a main challenge to provide privacy and security. The first approach towards the privacy and security of an RFID tag is to kill [7] the tag at the point of sale. Due to various reasons for which killing a tag is not expected because consumer may wish to reuse the tag. Weis et al. [8] proposed a hash-based access control (HAC) approach to protect a tag using a one-way hash function. To do it, the tag stores the hash of a random key as a metaID. Since the metaID is same for a tag all the times, it always transmits the same metaID, which can easily be tracked by an adversary [9]. Another problem in this system is that the information is transmitted in plain text which can easily be eavesdropped. Weis et al. [8] also suggested another approach that is extended from HAC called randomized access control [8]. It uses a random number to prevent location privacy. In each session, the tag produces a response with newly generated random number and its *ID* using a hash function. It cannot protect the system from replay attack and is not suitable in a real-life system where a large number of tags are used as it requires many expensive hash operations at the back-end database.

To protect location privacy several protocols use hash function of varied identifier or varied secret [9-13]. Chien and Chen [9] proposed a challenge-response-based authentication protocol to prevent replay attack. This protocol uses a database in the server which maintains new and old tag keys to protect DoS attack. To prevent traceability, authentication and access keys are updated. However, this scheme is still vulnerable to backward and forward traceability because if an active attacker compromise a tag this can identify the tag's past interactions from previous transactions and the fixed *ID* of the tag and can deduce the future transaction. Ohkubo et al. [10] proposed an RFID privacy scheme using a hash chain (HC) mechanism. This method uses two hash functions to protect the privacy and security. It is also not suitable in practical use because the back-end database requires a large number of HCs. Henrici et al. [11] proposed a scheme referred to the hash-based *ID* variation scheme (HIDV). It uses a one-way hash function to protect location privacy by changing the *ID* after each session. However, if any authentication session is unsuccessful it replies with the same hashed *ID* again for which it opens up the vulnerability for impersonation attack such as spoofing. Lee et al. [12] proposed a low-cost authentication protocol (LCAP) which simplifies and enhances HIDV scheme in both efficiency and security. It has the similar problem as in HIDV that a tag always replies with the same hashed *ID* before the next successful

authentication which allows tag tracking. Dimitriou [13] proposed an RFID authentication scheme that preserves user privacy and also protects against tag cloning. This protocol uses the hash of its identifier as a response to a reader query to maintain scalability at the server, and the back-end server sends a message using the updated identifier to the tag after getting the tag response. This scheme also has a problem of tracking as between valid sessions the tag identifier remains the same.

Varying identifier may cause problem in ubiquitous computing environment. To solve the problem, static identifiers are used in many authentication protocols. Molnar et al. [14] proposed a private authentication scheme for library RFID systems. It uses a pseudorandom number and a shared secret key by the tag and the reader for efficient authentication. This scheme does not ensure forward security since the tag's identifier and the secret key are static and the random number forwarded is in plain text which can be captured by an adversary. Rhee et al. [15] proposed challenge response-based RFID authentication protocol (CRAP) which is designed to use in ubiquitous computing. However, this scheme requires $(N/2 + 1)$ hash function computations in the database which is impractical for large number of tags in ubiquitous computing. Choi et al. [16] proposed a one-way hash-based LCAP (OHLCAP), which is suitable for ubiquitous environment. Ha et al. [17] claim that OHLCAP suffers from traceability and impersonation attack. The authors also proposed a solution of using hash function to protect from traceability attack. Tsudik [18] described an RFID identification protocol that provides a basic level of tag identification using time-stamps. Tsudik [18] also proposed two further schemes that provide tag authentication. The schemes use monotonically increasing time-stamps for tracking-resistant tag authentication, and employ a keyed hash function f .

Karthikeyan and Nesterenko [19] proposed RFID security protocol without computationally expensive cryptographic mechanisms and used simple matrix multiplication. However, this protocol is vulnerable to DoS attack and intruder can try a brute-force matrix and key guessing attack. Moreover, this scheme is not secure from replay and tracking attack. Song and Mitchell (SM) [20] proposed an RFID authentication protocol and ownership transfer protocol [21] to prevent all the attacks discussed so far. Although these protocols are efficient in terms of storage and computation requirements, but they are vulnerable to both tag impersonation attack and reader impersonation attack. Cai et al. [22] proposed a revised authentication protocol of SM [20] to eliminate the problems in it without violation of any other security properties. The storage and computation requirements are also comparable with the existing protocol.

In the next two sections, two prominent protocols LCAP [12] and OHLCAP [16] will be discussed in more detail as they are more related to the proposed work.

3.1 LCAP approach

LCAP scheme uses one-way hash function to protect the privacy and security of the tag. The notations and symbols used in LCAP operation are as follows [12]:

$h: \{0, 1\}^* \rightarrow \{0, 1\}^l$ is a one-way hash function

ID : ID denotes identity of a tag and is a random value in $\{0, 1\}^l$.

Data fields of a tag and a reader are initialized to the following values:

Tag: The data field of a tag is initialized to its own ID .

Reader: A reader picks uniformly a random number r in $\{0, 1\}^l$.

The data fields of a back-end database are initialized to $HaID$, ID , TD and $DATA$.

HaID: $HaID$ value is the hash value of ID used for identifying or addressing the tag.

TD: TD -entry is used to trace previous data information of a tag when loss of message occurs in the current session.

DATA: $DATA$ stores the information about an accessible tag.

The back-end database maintains two rows; *Prev* for the previous session and *Curr* for the current session. Each row contains $HaID$, ID , TD , and $DATA$ fields. In *Prev*, the back-end database records $HaID$ and ID in the previous session. In *Curr*, it updates $HaID$ and ID of *Prev*. TD -field of *Curr* has $HaID$ value of *Prev* and TD -field of *Prev* contains $HaID$ -value of *Curr*. The protocol is shown in Figure 2.

LCAP works as follows:

1. A reader selects a random number r and sends a *Query* and r to the tag.
2. The tag computes $HaID = h(ID)$ and $h(ID||r)$ using r and its ID and sends $h_L(ID||r)$ and $HaID$ to the reader, where $h_L(ID||r)$ is the left half of $h(ID||r)$.
3. The reader sends $h_L(ID||r)$, r , and $HaID$ to the back-end database.
4. The back-end database then compares if the value of $HaID$ in *Prev* is same as the value of $HaID$ received from the reader. If successful, then the back-end database computes $h_R(ID||r)$ using r received from the reader and ID in *Prev*, where $h_R(ID||r)$ is the right half of $h(ID||r)$. For the next session, the back-end database computes and stores $HaID = h(ID \oplus r)$ and $ID = ID \oplus r$ in *Curr*. TD -field of *Prev* is filled with current $HaID = h(ID \oplus r)$. Finally, the back-end database sends $h_R(ID||r)$ to the reader.
5. The reader forwards $h_R(ID||r)$ to the tag.
6. The tag checks $h_R(ID||r)$. If it matches, the tag updates its ID to $ID \oplus r$.

In LCAP scheme, ID is changed in each authentication. So, it does not work in ubiquitous environment. Also mentioned earlier, it cannot overcome the traceability problem.

3.2 OHLCAP approach

OHLCAP uses a static identifier and secrets and is suitable for ubiquitous environment. It also uses a one-way hash function for privacy and security of the tag. OHLCAP requires an ID and a hash function h as in LCAP. Some additional fields are also required. GI is used as a group index. K is a common secret used in all tags and S is a tag secret. The protocol computes three messages: $A^1 = K \oplus c$, $A^2 = ID + (GI_i \oplus r \oplus c) \bmod (2^l - 1)$, and $B = h(ID||(S \oplus GI_i)|(r \oplus c))$. B_L and B_R are the left and right half of B , respectively. c is used as a counter and initialized to an arbitrary value. It is increased every time a reader sends a query to the tag. The protocol is shown in Figure 3.

OHLCAP works as follows [16]:

Step 1: A reader selects a random value r and sends a query with r to a tag.

Step 2: The tag checks a random value r whether it is all zero value or not.

1. If r value is all zero, the tag sends "stop" message to the reader and stop the protocol.

2. Otherwise, the tag performs the following

- The tag computes $A^1 = K \oplus c$, $A^2 = ID + (GI_i \oplus r \oplus c) \bmod (2^l - 1)$,

$B = h(ID||(S \oplus GI_i)|(r \oplus c))$ and sends A^1 , A^2 and B_R to the reader, where B_R is a right half of B ,

- Then, the tag increases the counter c which should not exceed $2^l - 1$.

If the counter c exceeds $2^l - 1$, it is initialized by initial c .

Step 3. After receiving from the tag,

1. The reader forwards A^1 , A^2 , B_R and r to the back-end database.

2. The back-end database computes $c' = A^1 \oplus K$ and $ID_j' = A^2 - (GI_j \oplus r \oplus c') \bmod (2^l - 1)$ using all group indices GI_j , $j \in \{1, \dots, n\}$

3. The back-end database checks if one of computed $ID_j'_{j \in \{1, \dots, n\}}$ is matching to one of the stored ID s in the back-end database. If this process succeeds, the back-end database check if the GI_j used to compute the ID_j' is equal to the group index GI_i that contains the matching ID_j' .

- If this is successful, the database computes $B = h(ID||(S \oplus GI_i)|(r \oplus c))$ using the matched ID .

- Otherwise, the back-end database stops this process.

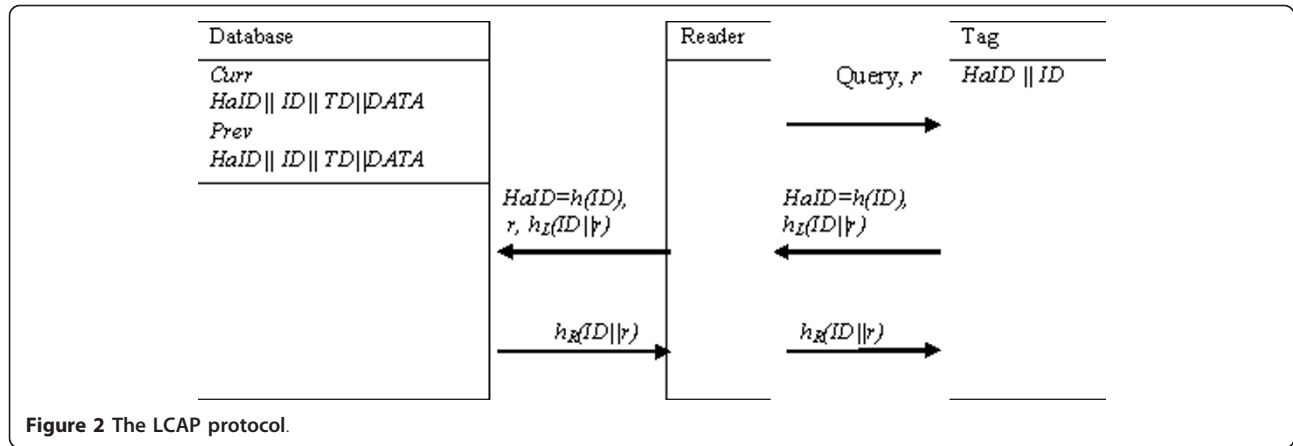


Figure 2 The LCAP protocol.

4. Then, the back-end database authenticates the tag by matching the received value B_R .

5. The back-end database sends B_L to the reader, where B_L is a left half of B . The reader forwards B_L to the tag.

Step 4. The tag authenticates the reader by comparing the received value B_L .

OHLCAP is an efficient approach in ubiquitous environment that uses one-way hash function for privacy and security. However, Ha et al. [17] find its security weakness and proposes an enhanced OHLCAP (EOHLCAP) scheme. The authors showed that this protocol is vulnerable to traceability attack and impersonation attack because of its special property, namely, $c_c = c_p + 1$ for two successive sessions. The adversary eavesdrops the messages transmitted between the tag and the reader and obtains the successive A_p^i and A_c^i where $A_p^i = K \oplus c_p$, $A_c^i = K \oplus c_c$. Afterwards, it computes $A = A_p^i \oplus A_c^i = c_p$

$\oplus c_c = c_p \oplus (c_p + 1)$ and removes the secret key K in this equation. In this way, the adversary can trace the tag's holder. Similarly, the adversary can implement impersonation attack by selecting special random number $r_c = r_p + 1$. If $r_c \oplus c_c = r_p \oplus c_p$ then the value of B_p is equal to B_c since $B = h(ID || (S \oplus GI_i) || (r \oplus c))$. To overcome the security weakness, Ha et al. [17] add a pseudorandom number generator to generate a random number and removes the counter in the tag to prevent traceability attack.

4. Our contributions: secure ubiquitous authentication protocols (SUAP)

OHLCAP is not protected against traceability and impersonation attacks. It requires much storage in the tag side and database side. EOHLCAP eliminates the privacy problem in OHLCAP with reduced amount of storages in the tag side and the database side but it takes many hash operations to locate the tag in the database [17]. LCAP

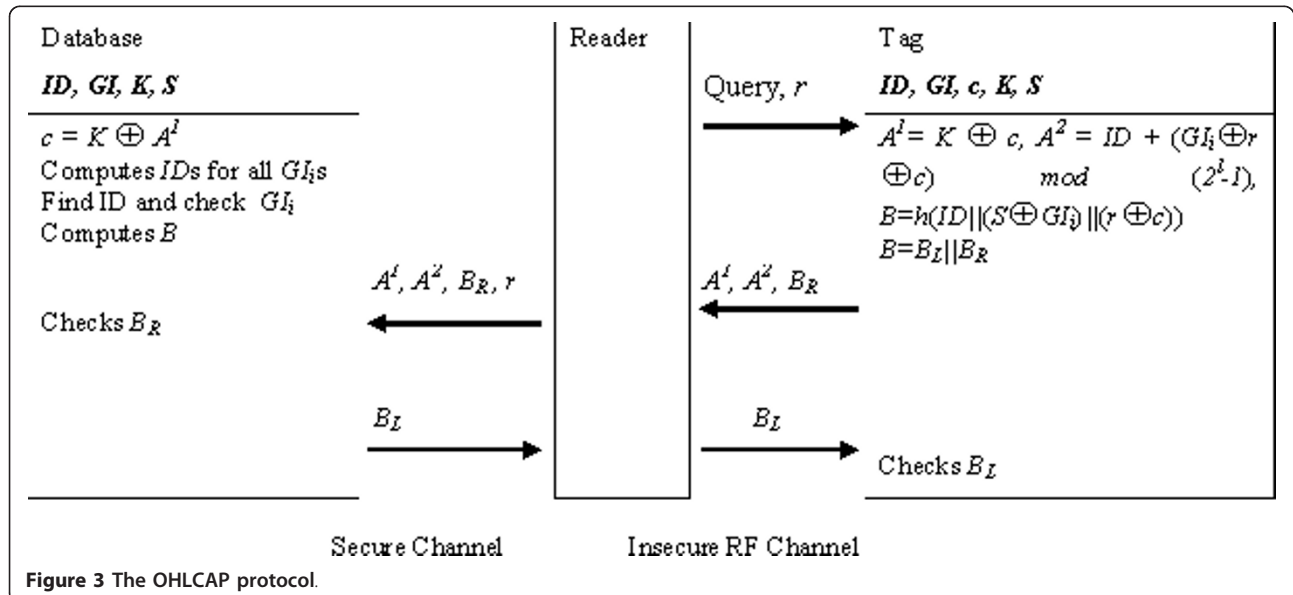


Figure 3 The OHLCAP protocol.

requires less storage in the tag and reduces search time in the database but it is not suitable for ubiquitous computing environment as the *ID* is updated after each authentication process. To overcome these problems, three Secure Ubiquitous Authentication Protocols SUAP1, SUAP2 and SUAP3 for RFID systems are proposed in this section. SUAP1 is a simple RFID authentication protocol that will work in a system where the number of tags is small. The preliminary version of SUAP1 is presented in [23]. The final version of SUAP1 is improved over the preliminary version with privacy and security enhancement. SUAP2 and SUAP3 are the extension of SUAP1 and work in a large group-based system where RFID tags are divided into several groups. These protocols are low-cost and secured based on challenge-response method using a one-way hash function, hash-address as a search index. The proposed protocols combine the features of the hash address and hash function of the LCAP protocol and the ubiquitous property of OHLCAP protocol and overcome the existing privacy and security problems in these two schemes. The advantage of hash address is to reduce the search time in the database. The notations used in the SUAP1, SUAP2 and SUAP3 protocols are as follows:

Notations

- h* A one-way hash function, $h: \{0,1\}^* \rightarrow \{0,1\}^l$
ID Tag identifier
GID Group identifier
Had Hash address $h(ID)$
N Number of tags
n Number of groups
m_i Number of tags in the *i*th group
l The length of an identifier. The value of *l* is assumed 96 bits.
r₁ Random number in $\{0,1\}^l$
r₂ Random number in $\{0,1\}^l$
 \oplus XOR operator
 \parallel Concatenation operator
 \leftarrow Assignment operator

4.1 SUAP1

SUAP1 protocol uses a static identifier and a secret number, hash functions and two random numbers. The objective of this protocol is to preserve the ubiquitous property of the protocol and is suitable for a small number of tags. In this case a common secret is stored in all the tags. Two random numbers make the hash response unpredictable so that it is impossible to perform impersonation and tracing attack by a malicious reader. The system set-up of SUAP1 protocol is as follows:

System Set-up

Tag: Each tag contains the following fields:

ID: Tag Identifier

x: Common secret number

Reader: Reader does not contain any field.

Back-end database: Back-end database contains the following fields:

ID: Tag identifier

x: Common secret number

Had: Hash address $h(ID)$

When a tag enters into the range of a reader, the reader can initiate the authentication protocol. The protocol is shown in Figure 4. The steps in the authentication protocol are as follows:

1. The reader generates a random number r_1 and sends it to the tag.
2. Receiving the number r_1 the tag generates another random number r_2 .

If r_1 or r_2 is 0 stop protocol

Otherwise, the tag performs the following computations

$$y \leftarrow h(ID) + (r_1 \oplus r_2)$$

$t = r_2 \oplus x$. *t* is a temporary variable here.

Computes $h(ID \parallel r_1 \parallel r_2)$

The tag then sends the value of *y*, *t* and h_L to the reader.

Where h_L is the left half of $h(ID \parallel r_1 \parallel r_2)$

3. The reader then sends the value of *y*, *t* and r_1 to the back-end database.
4. The back-end database will calculate the following.

$$r_2 = t \oplus x$$

$$h(ID) \leftarrow y - (r_1 \oplus r_2)$$

$h(ID)$ is the address of the record containing the *ID* where $Had = h(ID)$

Access the address *Had*

Retrieves the *ID* from the record

Then the back-end database computes $h(ID \parallel r_1 \parallel r_2)$

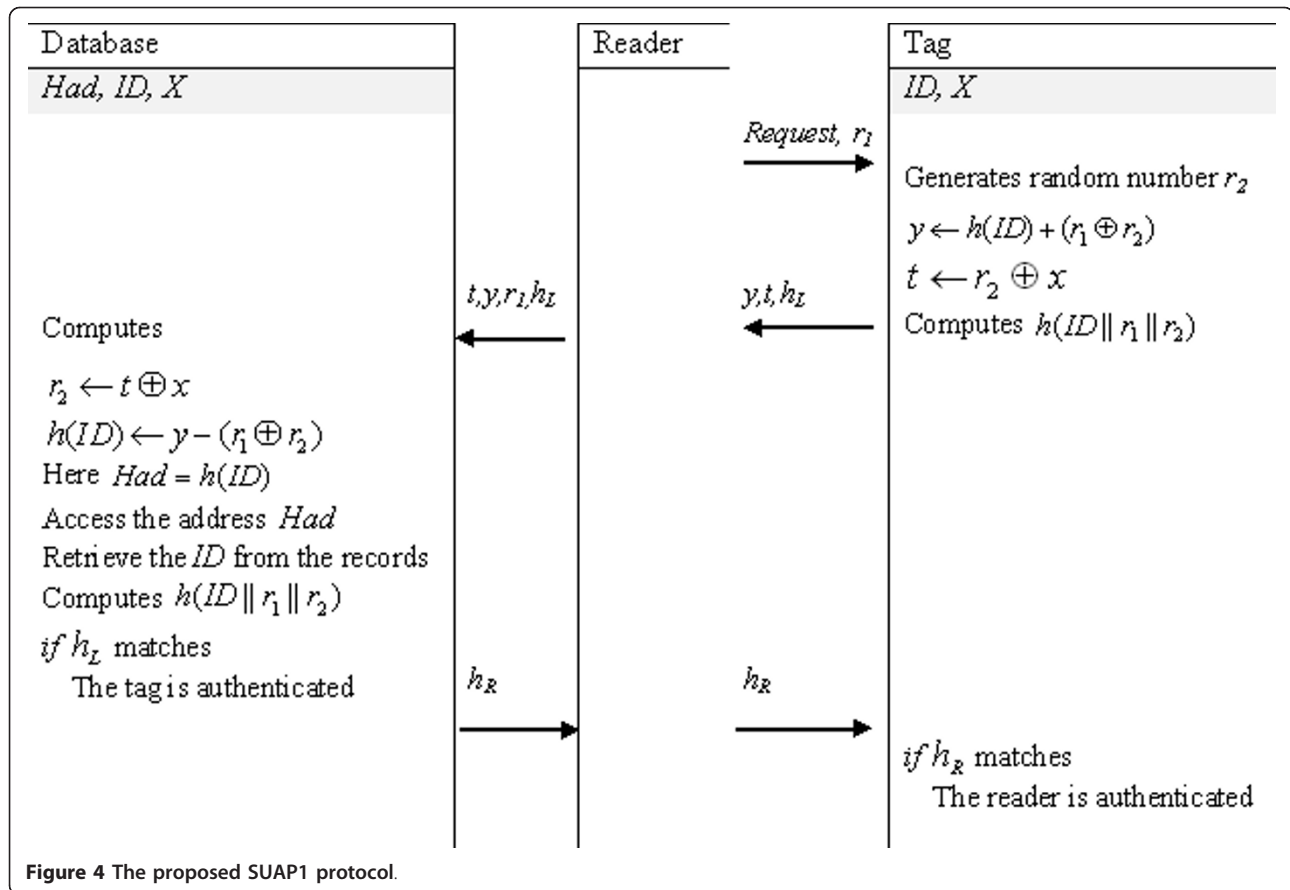
If h_L matches, the tag is authenticated

Sends h_R to the reader, where h_R is the right half of $h(ID \parallel r_1 \parallel r_2)$

5. The reader forwards the h_R to the tag

6. If the received h_R matches, the reader is authenticated.

The protocol is simple and works for an organization having small number of tags (i.e. several thousands). Two random numbers make the response anonymous. The problem in this simple protocol is that it maintains a common secret for all the tags in the database. It can be a problem to manage this secret in a large organization having different departments. Having only a single secret



x for all the tags it cannot ensure privacy and security for a large organization having millions of tags in many departments.

4.2 SUAP2

To overcome the problem in SUAP1 of having only one secret for all the tags, SUAP2 maintains groups for the different departments and different types of products. In addition to the ID and secrets in the SUAP1, one extra variable GID is needed in the tag side and the database side. It represents a group identifier. This is also a secret number. The database divides the tags into n groups and the protocol is shown in Figure 5. The only difference between the SUAP1 and SUAP2 is that SUAP2 maintains the groups of the tags and there is a common secret for each group. In this case one secret value x is used for all the tags in a group. It will reduce the tag search time in the database. This is suitable for the case where the tags of the same group are not distributed in various places. It ensures better security but requires less computation and search times in the database. The system set-up of SUAP2 protocol is as follows:

System Set-up

Tag: Each tag contains the following fields:

ID : Tag Identifier

x : Secret number for a group

GID : Group identifier

Reader: Reader does not contain any fields.

Back-end database: Back-end database contains the following fields:

ID : Tag identifier

x : Secret number for a group

Had : Hash address $h(ID)$

GID : Group identifier

The steps in the authentication protocol are as follows:

1. The reader generates a random number r_1 and sends it to the tag.

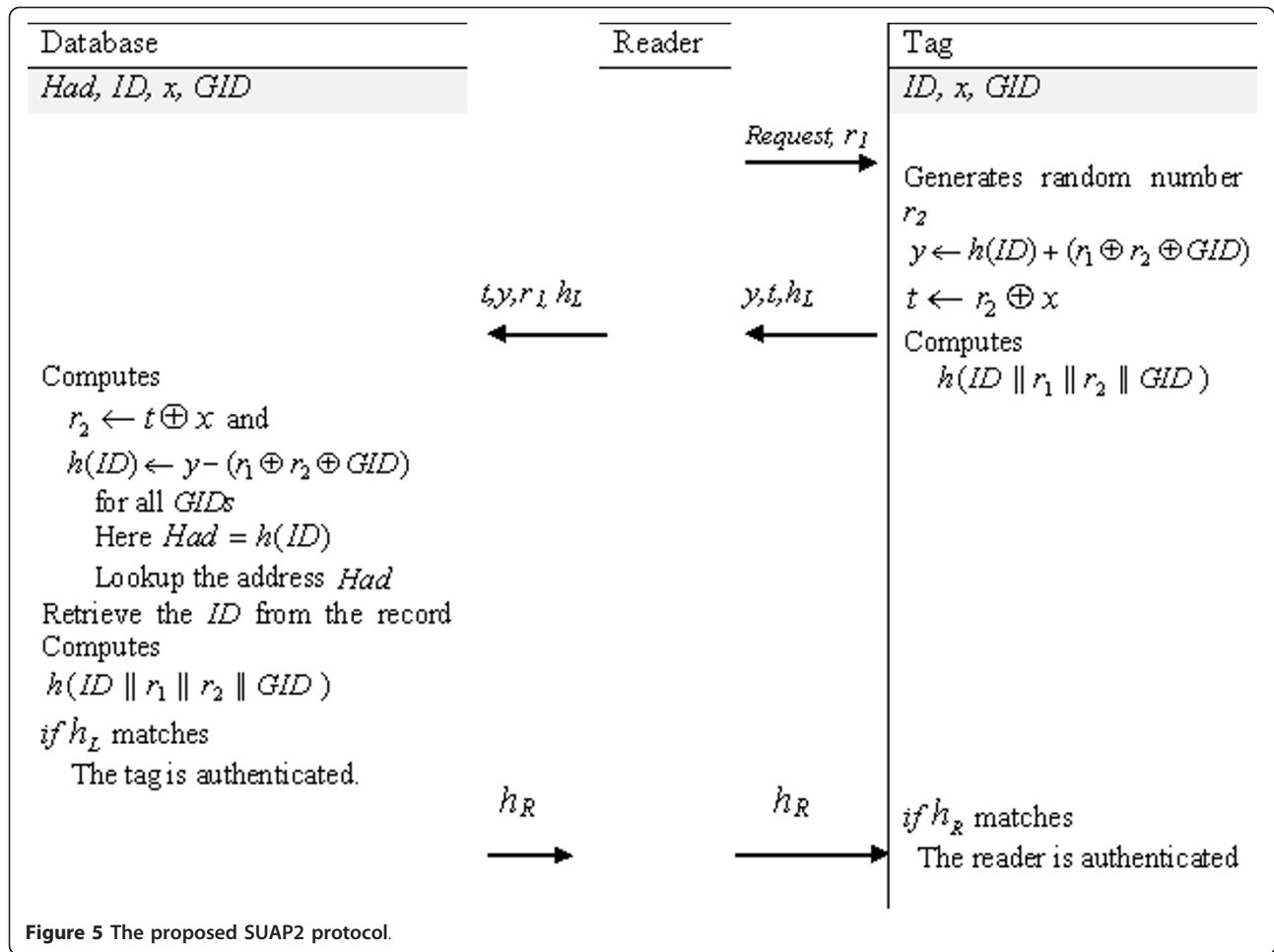
2. Receiving the number r_1 the tag generates another random number r_2 .

if r_1 or r_2 is 0 stop protocol

Otherwise, the tag performs the following computations

$$y \leftarrow h(ID) + (r_1 \oplus r_2 \oplus GID)$$

$$t = r_2 \oplus x$$



Computes $h(ID \parallel r_1 \parallel r_2 \parallel GID)$

The tag then sends the value of y , t and h_L to the reader.

Where h_L is the left half of $h(ID \parallel r_1 \parallel r_2 \parallel GID)$

3. The reader then sends the value of y , t and r_1 to the back-end database.

4. The back-end database calculates the following for all $GIDs$

$$r_2 = t \oplus x$$

$$h(ID) \leftarrow y - (r_1 \oplus r_2 \oplus GID)$$

$h(ID)$ is the address of the record containing the ID where $Had = h(ID)$

Lookup the address Had

Retrieves the ID from the record

Then the back-end database computes $h(ID \parallel r_1 \parallel r_2 \parallel GID)$

If h_L matches, the tag is authenticated

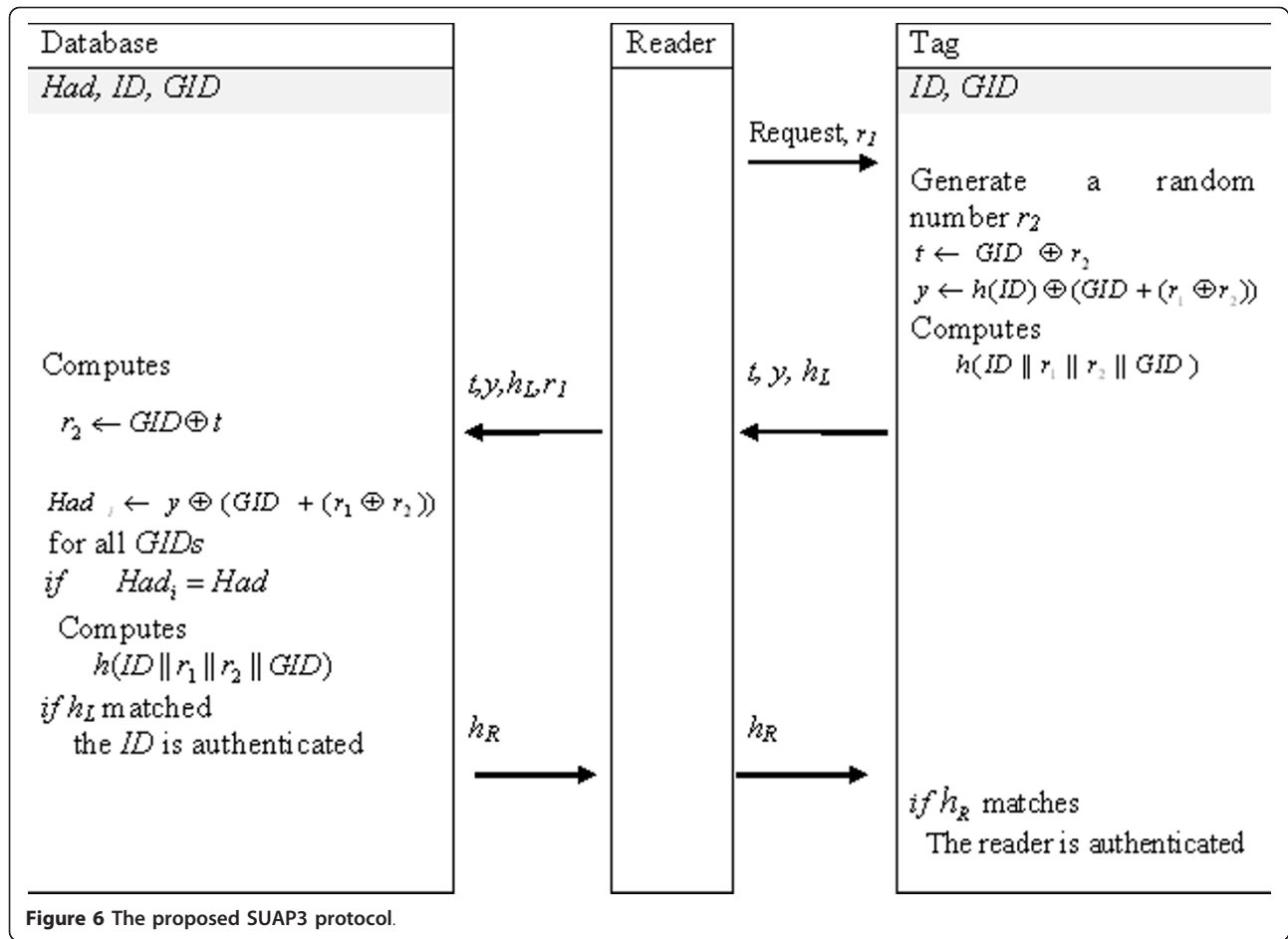
Sends h_R to the reader, where h_R is the right half of $h(ID \parallel r_1 \parallel r_2 \parallel GID)$

5. The reader forwards the h_R to the tag

6. If the received h_R matches, the reader is authenticated.

4.3 SUAP3

The SUAP3 enhances the SUAP2 in efficiency by removing the secret x from the tag and the database. It keeps the group variable GID in the tag and the database as in SUAP2. It represents a group identifier and also a secret number. The database divides the tags into n groups and the protocol is shown in Figure 6. The only difference between the SUAP2 and SUAP3 is that SUAP3 does not use the secret x for the tag and the database. The group-based structure is used for the searching tags in the database. The privacy will not be hampered due to the elimination of the secret x as the GID works as an l bits secret which is also difficult to guess by the adversary. It reduces the number of searches significantly. Since the hash function is one-way it still gives the same security protection



to the *ID*. The system set-up of SUAP3 protocol is as follows:

System Set-up

Tag: Each tag contains the following fields:

ID: Tag Identifier

GID: Group identifier

Reader: Reader does not contain any fields.

Back-end database: Back-end database contains the following fields:

ID: Tag identifier

Had: Hash address $h(ID)$

GID: Group identifier

The steps in the authentication protocol are as follows:

1. The reader generates a random number r_1 and sends it to the tag.

2. Receiving the number r_1 the tag generates another random number r_2 .

if r_1 or r_2 is 0 stop protocol

Otherwise, the tag performs the following computations

$$t \leftarrow GID \oplus r_2$$

$y \leftarrow h(ID) \oplus (GID + (r_1 \oplus r_2))$ and computes $h(ID || r_1 || r_2 || GID)$

The tag then sends the value of y , t and h_L to the reader.

Where h_L is the left half of $h(ID || r_1 || r_2 || GID)$

3. The reader then sends the value of y , r_1 and t to the back-end database.

4. The back-end database calculates the following for all *GIDs*

$$r_2 \leftarrow GID \oplus t$$

$$Had_i \leftarrow y \oplus (GID + (r_1 \oplus r_2))$$

Had_i is the address of the record containing the *ID* where $Had_i = h(ID)$

Lookup the address Had_i in the database

If $Had_i = Had$ for any ID retrieves the ID from the record

Then the back-end database computes $h(ID||r_1||r_2||GID)$

If h_L matches, the tag is authenticated

Sends h_R to the reader, where h_R is the right half of $h(ID||r_1||r_2||GID)$

5. The reader forwards the h_R to the tag

6. If the received h_R matches, the reader is authenticated.

5. Evaluation of the proposed protocols

The protocols are analysed in two ways: first is the privacy and security analysis and the second one is the efficiency analysis.

5.1 Privacy and security analysis

The privacy and security of the proposed protocols are analysed against various threats introduced in Section 2; information leakage of a tag, location privacy, impersonation and replay attack, DoS attack and traceability. The privacy and security analysis against the identified threats is outlined as follows:

- **Information leakage:** In SUAP1 protocol, the adversary must be authenticated to access any sensitive information from a tag. To authenticate the systems an adversary must know ID , x and r_2 to access any information from the tag. The SUAP2 protocol has additional GID secret to make the response more unpredictable. The SUAP3 uses the GID as a secret instead of x . The combination of r_1 and r_2 makes the response y so unpredictable that the adversary can only guess the value of h_R and h_L . The advantage of an adversary is at most $\frac{1}{2^l}$, which is negligible for $l = 96$ or more.

- **Location privacy:** The responses from the tags are always changing in every new session. The value of t , y and h_L cannot be linked with any particular tag in SUAP1, SUAP2 and SUAP3. The protocols ensure location privacy by using new values of r_1 and r_2 each time. Even if a malicious reader sends the same random value r_1 all the times, a tag transmits the refreshed values that are refreshed by r_2 and secret value x or GID .

- **Impersonation and replay attack:** The protocols work in a complete challenge-response fashion by mutual authentication. When a tag reaches within the range of a reader, the reader sends queries with a random number to the tag. An adversary may also request a tag with a random number. Without knowing ID , hash function, secret x and random number r_2 generated by the tag, the adversary cannot find the response y . In SUAP2 and SUAP3, the group identifier GID also makes the response more unidentifiable. For each session the tag gives a new value of y that is totally indistinguishable and different

from other sessions. So, impersonation and replay attack is nearly impossible in practical scenario. Impersonation and replay attack could be possible if the attacker waits for a matched response (same h_L) from the tag and replays the h_R to authenticate itself. Such repeating hash response could only be reproduced once in 2^l responses (where the responses are uniformly random in nature) as the length of the hash response is l .

- **DoS:** Since the ID and the secret are never changed in the proposed protocols, if the attacker prevents the last flow to the tag from the reader it will not cause any problem of desynchronization.

- **Traceability:** The schemes SUAP1, SUAP2 and SUAP3 are fully protected from future forward and backward traceability. The attacker has no access over r_2 , and the combination of r_1 and r_2 and the hash function. The responses are always anonymous and the attacker does not know the value of the ID and the secret. So the previous, present and future interactions are all indistinguishable. The attacker cannot identify the past and future interactions.

5.2 Efficiency analysis

For efficiency analysis the storage, communication and computation cost of the proposed protocols are compared with other protocols in Table 1. The storage cost indicates the storage requirements in the tag, database and the reader. The communication cost means the length of bits the tag and the reader send during the authentication process. The computation cost is the maximum computations require in the tag and the database during the execution of the authentication protocol.

In Table 1, LCAP [12] performs better than other protocols for almost all criteria but it suffers from traceability problem and it is not ubiquitous. The proposed protocols show better performance because it requires less tag side and database side storage and gives protection from all the known attacks. The storage requirement for the tag and the database are $2l$ and $3l$, respectively, in SUAP1 and SUAP3, whereas OHLCAP [16] requires $5l$ and $4l$, respectively. CRAP [15] uses only $1l$ storage for a tag but it needs $(N/2+1)$ hash operations which is practically unsuitable because in ubiquitous environment the value of N is extremely high and it does not divide the tags into groups. It requires many hash operations and hence requires long search time to obtain the tag information in the database. Similarly, the EOHLCAP [17] requires $3l$ storages in the tag side and $3l$ storages in the database side but requires a large number of hash operations for a group. This is also high for a group having a large number of tags. The main computation costs in the tags are the hash operations. OHLCAP requires 1 hash operations and additional operations A_l which are four XOR operations in the tag. EOHLCAP also requires 1

Table 1 Efficiency analysis

Efficiency criteria		LCAP	CRAP	OHLCAP	EOHLCAP	Proposed protocols		
						SUAP1	SUAP2	SUAP3
Storage	Tag	1/	1/	5/	3/	2/	3/	2/
	Reader	-	-	-	-	-	-	-
	Database	6/	1/	4/	3/	3/	4/	3/
Computation	Tag	2h	3h	1h(+A ₁)	1h(+A ₂)	2h (+A ₂)	2h (+A ₃)	2h (+A ₃)
	Reader	-	-	-	-	-	-	-
	Database	1h	$(\frac{N}{2} + 1)h$	1h+ε ₁	$(\frac{m_i + 1}{2})h + \varepsilon_2$	1h+ ε ₃	1h+ ε ₄	1h+ ε ₅
Communication	Tag-to-Reader	1.5/	2/	2.5/	2.5/	2.5/	2.5/	2.5/
	Reader-to-tag	0.5/	0.5/	0.5/	0.5/	0.5/	0.5/	0.5/

A₁, A₂, A₃, additional XOR and add operations in the tag; ε₁, ε₂, ε₃, ε₄, ε₅, small operations in the database

hash operation and additional operations A₂ which are two XOR operations in the tag. The proposed protocols require two one-way hash operations in each tag. SUAP1 requires additional operations A₂ which are two XOR operations. Both SUAP2 and SUAP3 require additional operations A₃ which are three XOR operations in the tag. In each protocol, the tag requires one addition operation. Since both XOR-operation and addition operation are very simple bit operation, hardware embodiment of these operations is simpler than one-way hash function. Therefore, the proposed protocols are suitable to a low-cost RFID tag systems. ε₁, ε₂, ε₃, ε₄ and ε₅ are additional operations other than hash functions in the corresponding databases as shown in Table 1.

6. Simulation experiment and evaluation

To validate the proposed protocols, simulation experiments have been conducted. The privacy and security protections are ensured with the hash functions and random numbers. A hash function is a one-way function for which the possibility of information leakage is negligible from the hash response. In the simulation experiment *l* takes different values, i.e. 16, 32, 64 and 96. However, many combinations of the hash inputs can give the same response that can be used by the adversary to impersonate the RFID systems through replay attack. This is the main reason to conduct the simulation. The objective of the simulation program is to verify the protection for impersonation, replay attack and location privacy. It checks the response *y* if it recurs more than once for one tag during the attacks by an adversary in a given number of attempts. If the same response is generated for any given random number pair it can be used by the adversary for impersonation and replay attack and the location privacy of the tag may be broken.

We simulate the impersonation and replay attack using Monte Carlo simulation method. To replay the hash value *h* (*h_L* || *h_R*) for a particular *ID* and *GID*, hash responses are generated for 10¹¹ times with the same *ID*

and *GID* and different set of *r₁* and *r₂*. The hash value generated at *i*th attempt *h_i* is considered vulnerable for impersonation and replay attack if *h_i* = *h*. The generated random sequences for *r₁* and *r₂* are tested for uniform random distribution using chi square test to ensure the validity of the simulation using Monte Carlo method. The number of matches found is recorded to generate the performance results. For a particular data length ten simulations are executed using different set of random numbers and the possible impersonation and replay attacks are observed in the simulation. The averages of the successful replay attacks are reported in Table 2.

The output of a hash function is the same for the same random number pair. Some different random number pairs may also give the same response. The objective is to ensure unique response for different inputs of random number pair so that an adversary is unable to use any response at later stage to access the tag or the reader. We select one tag and generate a response for two random numbers as in SUAP1, SUAP2, SUAP3 and EOHLCAP. Then the program attempts 10¹¹ times to check that how many times the same response is generated. This is the role of an adversary. In each attempt a new response is generated with a new pair of random number. The average number of times a similar response generated in SUAP1, SUAP2, SUAP3 and EOHLCAP are given in Table 2. The expected number of matches are also reported in a column to compare the obtained result. The value of the expected number of matches is calculated using the analysis of repeating hash response presented for replay attack in Section 5.1 and it is calculated as 10¹¹/2^{*l*}. All the selected protocols show almost similar results. The number of matches represents the success of the adversary to attack the tag. The experiment was conducted for 16, 32, 64 and 96 bits of secret value, random number, *ID* and hash response. The success of the adversary was found for 16 and 32 bits since many occurrences of the same response are found. There was no recurrence of the same response for 64 and 96 bits for the specified

Table 2 Attacker's success for one tag

Number	Number of attempts	Data length / (bits)	Expected number of matches	Average number of matches (attacker's success)			
				EOHLCAP	SUAP1	SUAP2	SUAP3
1	10^{11}	16	1525878.91	1532979.81	1536442.83	1535009.84	1526520.77
2	10^{11}	32	23.28	21.34	20.30	21.20	20.81
3	10^{11}	64	5.42×10^{-9}	0	0	0	0
4	10^{11}	96	1.26×10^{-18}	0	0	0	0

Table 3 Privacy and security comparisons

Property	LCAP	CRAP	OHLCAP	EOHLCAP	YA_TRAP*	SUAP1	SUAP2	SUAP3
Information privacy	Y	Y	Y	Y	Y	Y	Y	Y
Location privacy	N	Y	Y	Y	Y	Y	Y	Y
Impersonation	A	Y	N	Y	Y	Y	Y	Y
Replay attack	Y	Y	N	Y	Y	Y	Y	Y
Message interception	Y	Y	Y	Y	N	Y	Y	Y
Backward traceability	Y	Y	N	Y	N	Y	Y	Y
Forward traceability	Y	Y	N	Y	N	Y	Y	Y

Y, provided; A, provided under assumption; N, not provided

number of attempts, i.e. 10^{11} times. We did not perform simulation experiments for LCAP, OHLCAP and YA_TRAP* protocols since these are not protected against all the privacy threats [16-18]. CRAP is also not included since it requires many hash operations [16].

The simulation program has been developed using Turbo C++ compiler and the experiment was conducted in a desktop computer with 2.93GHz Intel (R) Core 2 Duo Processor, 3.46 GB memory and Windows XP professional Operating System.

According to the privacy and security analysis in Section 5.1 and the simulation results the summary of the privacy and security properties are given in Table 3.

The privacy and security properties of the proposed protocols are compared with five other schemes. The five schemes were chosen because all of these protocols involved tag authentication. LCAP involves secret update but other four protocols CRAP, OHLCAP, EOHLCAP and YA_TRAP* do not support secret update. Proposed protocols are more similar to CRAP, OHLCAP, EOHLCAP and YA_TRAP* than LCAP since all these protocols support authentication in ubiquitous computing environment and do not update the identifier and secret value. Table 3 shows that the proposed protocols provided protections from all the identified privacy and security threats.

7. Conclusion

Three efficient and secure authentication protocols SUAP1, SUAP2 and SUAP3 are proposed to protect privacy and security for the low-cost RFID system in ubiquitous computing environment. The privacy and security problems of LCAP and OHLCAP are overcome in these

protocols. SUAP1 is suitable for the organization having small number of tags. SUAP2 and SUAP3 are suitable for medium and large organizations having many departments. All the proposed schemes require only two one-way hash function operations and avoid large number of hash computations in the database and hence are very efficient. The tag search time in the database is reduced by using the hash value as the address of the corresponding tag. EOHLCAP also overcomes the problem in OHLCAP and protects the RFID system from most of the attacks but it requires many complex hash operations. The proposed protocols ensure privacy and security protections from all the identified threats. The storage requirements in SUAP1 and SUAP3 are also less than OHLCAP and EOHLCAP protocols. The comparison shows that the proposed protocols are both secure and efficient than other schemes and have practical advantages over them because these are simple and provide a larger range of privacy and security protections for low storage and computations.

Competing interests

The authors declare that they have no competing interests.

Received: 14 July 2011 Accepted: 8 March 2012

Published: 8 March 2012

References

1. A Jules, S Garfinkel, R Pappu, RFID privacy: an overview of problems and proposed solutions. *IEEE Security Privacy*. **3**(3), 34-43 (2005). doi:10.1109/MSP.2005.78
2. A Jules, RFID security and privacy: a research survey. *IEEE J Sel Areas Commun*. **24**(2), 1-19 (2006)
3. R Want, An introduction to RFID technology. *IEEE Pervasive Comput*. **5**, 25-33 (2005)

4. BS Prabhu, X Su, H Ramamurthy, C Chu, R Gadh, WinRFID-a middleware for the enablement of radio frequency identification (RFID) based applications UCLA, in *Wireless Internet for the Mobile Enterprise Consortium (WINMEC)*, Los Angeles, CA, 1–23(2003)
5. http://www.EPCglobalinc.org. EPCglobal Web site, 2005. Referenced 2005
6. S Sarma, S Weis, D Engels, Radio-frequency identification: security risks and challenges. *CryptoBytes*. **6**(1), 2–9 (2003)
7. A Juels, RL Rivest, M Szudlo, The blocker tag: selective blocking of RFID tags for consumer privacy, in *the 8th ACM Conference on Computer and Communications Security*, (ACM Press, Washington DC, USA, 2003), pp. 103–111
8. SA Weis, SE Sarma, RL Rivest, DW Engels, Security and privacy aspects of low-cost radio frequency identification systems, in *Security in Pervasive Computing*, vol. 2802. (Lecture Notes in Computer Science, 2004), pp. 201–212. doi:10.1007/978-3-540-39881-3_18
9. H Chien, C Chen, Mutual authentication protocol for RFID conforming to EPC class 1 generation 2 standards. *Comput Standard Interface*. **29**(2), 254–259 (2007). doi:10.1016/j.csi.2006.04.004
10. M Ohkubo, K Suzuki, S Kinoshita, Cryptographic approach to "privacy-friendly" tags, in *RFID Privacy Workshop, MIT, MA, USA* http://rfidprivacy.media.mit.edu/2003/papers/ohkubo.pdf (November 2003)
11. D Henrici, P Muller, Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. in *International Workshop on Pervasive Computing and Communication Security - PerSec 2004*, ed. by Sandhu R, Thomas R IEEE Computer Society, Orlando, FL, USA 149–153 (March 2004)
12. SM Lee, YJ Hwang, DH Lee, JI Lim, Efficient authentication for low-cost RFID systems, in *ICCSA'05, LNCS*, vol. 3480. (Springer-Verlag, 2005), pp. 619–629
13. T Dimitriou, A lightweight RFID protocol to protect against traceability and cloning attacks, in *Conference on Security and Privacy for Emerging Areas in Communication Networks - SecureComm*, IEEE Athens, Greece, pp. 59–66 (September 2005).
14. D Molnar, D Wagner, Privacy and Security in Library RFID: Issues, Practices, and Architectures, in *Conference on Computer and Communications Security*, ed. by Pfitzmann B, Liu P (ACM CCS, Washington, DC, USA, 2004), pp. 210–219. ACM Press
15. K Rhee, J Kwak, S Kim, D Won, Challenge-response based RFID authentication protocol for distributed database environment. *SPC 2005, LNCS 3450* 70–84 (2005)
16. EY Choi, SM Lee, DH Lee, Efficient RFID authentication protocol for ubiquitous computing environment. *Embed Ubiquit Comput*. **3832**, 945–954 (2005)
17. J Ha, S Moon, JMG Nieto, C Boyd, Security analysis and enhancement of one-way hash based low-cost authentication protocol. *Emerging Technol Knowl Disc Data Mining*. **4819**, 574–583 (2007). doi:10.1007/978-3-540-77018-3_57
18. G Tsudik, A family of dunces: trivial RFID identification and authentication protocols, in *7th International Symposium on Privacy Enhancing Technologies-PET 2007, Lecture Notes in Computer Science*, vol. 4776, ed. by Borisov N, Golle P (Ottawa, Canada, 2007), pp. 45–61. Springer-Verlag, Berlin
19. S Karthikeyan, N Nesterenko, RFID security without extensive cryptography, in *Workshop on Security of Ad Hoc and Sensor Networks - SASN'05*, (Alexandria, Virginia, USA, 2005), pp. 63–67. ACM Press
20. B Song, CJ Mitchell, RFID authentication protocol for low-cost tags, (WiSEC'08, Alexandria, Virginia, USA, 2008), pp. 140–147
21. B Song, RFID tag ownership transfer, in *4th Workshop on RFID Security, RFIDsec08*, (Budapest, Hungary, 2008), p. 16
22. S Cai, Y Li, T Li, RH Deng, Attacks and improvements to an RFID mutual authentication protocol and its extensions, in *WiSec'09*, (Zurich, Switzerland, 2009), pp. 51–58
23. MM Morshed, H Yu, A Atkins, SI Ahamed, MM Akbar, A two-way RFID authentication protocol in pervasive computing, in *The 16th International Conference on Automation and Computing (ICAC'10)*, (Birmingham University, UK, 2010), pp. 164–169

doi:10.1186/1687-1499-2012-93

Cite this article as: Morshed et al.: Secure ubiquitous authentication protocols for RFID systems. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:93.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com